云—边缘系统中跨域大数据作业调度技术研究 *

徐　超[1]，吴　波[1]，姜丽丽[2]，金熠波[3]，张　胜[3]

(1. 国网江苏省电力有限公司电力科学研究院，南京 210008; 2. 江苏方天电力技术有限公司，南京 211102; 3. 南京大学 计算机科学与技术系，南京 210023)

摘　要：为提升用户服务质量，各类边缘集群部署于用户周围，在成为云数据中心重要补充的同时，也因其与用户不断交互而产生大量用户数据。为了降低因处理这些跨域大数据带来的作业完成时延，首先提出了以最小化系列跨域作业平均完成时间为优化目标的在线随机调度算法 ranTA。ranTA 基于跨域资源的异构性在线地计算出各计算任务调度至不同位置的偏好，并以此偏好作为概率调度每个计算任务；更进一步，为了避免将“热点”数据积压在边缘集群造成性能瓶颈，提出基于 ranTA 的捎带式数据重分布机制 ranTA-data，其将部分数据随任务执行留存至云数据中心。ranTA-data 不仅优化了当前作业的完成时间，也能证明在该机制下系列作业的平均完成时间以大概率汇聚于最优解附近。大规模仿真实验表明，所提出的在线随机化算法与数据重部署机制相比传统方法平均降低系列作业完成时间近 30%。

关键词：跨域数据处理；云—边缘集群；任务调度

中图分类号：TP311　　doi: 10.3969/j.issn.1001-3695.2018.08.0629

Task scheduling for geo-distributed data analytics in cloud-edge system

Xu Chao[1], Wu Bo[1], Jiang Lili[2], Jin Yibo[3], Zhang Sheng[3]

(1. *Research Institute* of *State Grid Jiangsu Electric Power Co. Ltd.*, *Nanjing* 210008, *China*; 2. *Jiangsu Frontier Electric Technology Co. Ltd.*, *Nanjing* 210000, *China*; 3. *Dept. of Computer Science & Technology*, *Nanjing University*, *Nanjing* 210023, *China*)

Abstract: Nowadays, many geo-distributed nearby edges have been deployed for providing high quality services to end users, which continuously produce large volume of user data. In order to minimize the average latency for a series of geo-distributed data analytical jobs, this paper first introduced online randomized algorithm ranTA. ranTA actually showed the preference on the task assignment under the consideration of both computing capacity of edges and the network bandwidth. Furthermore, in order to avoid overloading those edges with low computing capacities, this paper proposed data redistribution mechanism ranTA-data based on ranTA by redistributing some data to the central data center along with the tasks. The result of ranTA-data could be proved concentrated on its optimum with high probability. Extensive simulations show that ranTA-data gains nearly 30% improvement compared with current scheduling algorithms.

Key words: geo-distributed data analytics; cloud-edge system; task scheduling

0　引言

谷歌和阿里巴巴等大型企业与组织，已经在全球范围内部署了多个数据中心以及大量跨地域分布的边缘集群[1]。利用数据中心强大处理能力与边缘集群低时延的优势，这样的云-边缘系统为用户提供了高质量的业务，并且在各个边缘积累了大量用户数据[2]。而许多商业决策或数据分析需要实时综合处理这些跨域分布的数据[3]，因此如何在云-边缘系统中实现低时延的跨域大数据处理作业成为一个重要研究问题。

由于广域网数据传输的局限，将大量边缘数据先汇聚到云数据中心再处理的方式，不仅消耗带宽，也带来了较大的时延。有不少工作考虑尽可能将任务本地化执行，以减少广域网数据传输。Vulimiri 等人[4]研究如何在跨域环境下进行最少数据量的传输和快速任务执行；Pu 等人[5]发现利用稀缺带宽进行大规模数据传输容易造成各异的跨域传输时间，因此通过合理任务调度最小化跨域数据传输量。文献[6,7]在进行大数据处理作业的执行模式选择上也将数据传输与带宽的使用考虑在内，从而选出最优数据传输策略。然而，由于边缘集群在计算能力上的异构，纯粹优化数据传输的任务调度也会导致负载不均，造成一些任务在“热点”边缘积压。为此，文献[8,9]针对跨域环境带宽与计算力的异构性，提出了利用空闲资源与带宽进行批量任务调度，以减少批量任务的整体完成时间，但是一味地在本地计算资源被占用时将任务直接调度到远端云数据中心使用空闲计算资源，会给跨域链路带宽造成极大的负担。由于一些任务在本地进行适当的排队就能够获取到空闲的计算资源，为此 Jin 等人[10]针对空闲资源与占用资源使用不均衡的问题，设计了支持任务本地排队的批量任务调度方案，进一步降低批量任务的整体完成时延。

然而，所有这些研究工作都只针对当前提交的作业，通过任务调度来降低该作业完成时间。事实上，在云—边缘这

收稿日期：2018-08-27；**修回日期：**2018-10-11　**基金项目：**国家自然科学基金资助项目（61502224，61872175）

作者简介：徐超（1989-），男，山东莱芜人，工程师，硕士，主要研究方向为分布式系统、大数据处理（mmxcan@163.com）；吴波（1974-），男，江苏淮安人，工程师，本科，主要研究方向为分布式系统；姜丽丽（1989-），女，天津人，工程师，硕士，主要研究方向为边缘计算；金熠波（1994-），男，浙江上虞人，博士研究生，主要研究方向为分布式大数据处理系统；张胜（1986-），男，江苏镇江人，讲师，博士，主要研究方向为分布计算与并行处理.

样的异构分布式系统中，数据分布是影响作业执行的关键。如果能将“热点”数据尽可能转移到具有强大处理能力的数据中心，那么后续相关作业就可以高效完成。现有工作虽然优化了当前任务的完成时间，但并未考虑多个作业的平均完成时间，也就是，没有系统化研究由于当前任务调度引起的数据重部署对后续作业带来的收益。为此，本文以优化系列作业平均完成时间为目标，深入研究了跨域大数据作业的任务分配问题，提出了在线随机化任务分配算法 *ran*TA 与捎带式数据重部署策略 *ran*TA-*data*。不仅优化了当前作业的完成时间，也能证明系列作业的平均完成时间以大概率汇聚于最优解附近。大量模拟实验亦表明，在线随机化任务调度算法与捎带式数据重部署策略具有良好性能，相比传统方法在作业平均完成时间上降低近 30%。

1　云—边缘系统中跨域大数据处理

1.1　云—边缘系统与大数据处理

云—边缘系统中包含一系列跨域部署在各地的边缘集群，以及一个能力强大的云数据中心，对边缘集群以及云数据中心计算能力，一般采用计算单元[2]（slot）的数量来刻画。一般认为，能力强大的云数据中心计算单元数量总是充足的，而边缘集群相对于云数据中心的处理能力也较弱，计算单元数量有限，相关计算单元负载可能较高。所有边缘集群与云数据中心有网络连接，因此可以与云数据中心进行数据传输与信息交换。

一般的，数据分析作业可以定义为一个有向无环图（DAG）[5]，其中节点表示阶段功能，边表示阶段间的依赖关系。在运行过程中，主流的 Hadoop 和 Spark 等大数据处理平台针对 DAG 型作业的每个阶段，生成可并行执行的一组任务。同时，这些平台根据当前作业阶段，批量执行任务。由于 DAG 型作业中，一个阶段的所有任务全部完成，才能进入下一个阶段，因此，作业完成时间取决于最后一个完成的任务。对于一个计算单元 s，分配其上的任务串行执行，设 w_s 表示一个计算单元 s 上的待处理负载，即为计算单元 s 上处理完所有等待任务的时间。每一个数据分析任务均使用一个存储于 HDFS [11]且大小为τ的数据块作为输入数据进行处理[12]，因此，数据必须传送到运行任务的计算单元。边缘集群计算单元数量有限，可能导致任务排队而延长任务执行时间；数据中心计算资源充足，任务能快速执行，但往往面临从边缘抽取数据消耗时间。云-边缘系统中作业调度目标就是，在大数据处理平台开始执行某阶段一批任务前，决定这些任务应该在保存相关数据的边缘集群执行还是转移到数据中心，从而最小化作业的完成时间。

由于一部分数据分析任务留在本地集群执行，另一部分至远端数据中心进行处理，因此需要评估任务调度后批量数据分析任务在本地与数据中心产生的负载，以优化作业完成时间。对于某一个数据分析作业 j，及其包含的一批数据分析任务，设这些任务将会在边缘集群 i 上访问的数据集为 T_i^j，那么单个作业优化的目标即为最小化批量任务的完成时间。而批量任务产生的负载包含两个方面：a）在本地边缘集群产生的计算负载，设 U_s^j 表示作业 j 调度后计算单元 s 上的总负载；b）将任务调度到云数据中心产生的负载，设 V_i^j 为作业 j 调度后边缘集群 i 向云数据中心转移的负载，同时设对于作业 j 而言边缘集群 i 至云数据中心的可用带宽为 B_i^j 。因此，对于任何的计算单元 s，该两部分的负载可分别表示为

$$U_s^j = w_s^j + \gamma_{\phi(s)} \sum_{d \in T_{\phi(s)}^j} I_{ds}^j e_d^j \tag{1}$$

$$V_i^j = \frac{\tau}{B_i^j}\left(|T_i^j| - \sum_{d \in T_i^j, \phi(s)=i} I_{ds}^j\right) + \max_{d \in T_i^j}\left\{\left(1 - \sum_{\phi(s)=i} I_{ds}^j\right) e_d^j\right\} \tag{2}$$

其中: I_{ds}^j 为调度指示变量，用于表示作业 j 中以数据块 d 为输入的数据分析任务是否调度到计算单元 s 上。式（1）中的 e_d^j 表示为以数据块 d 作为输入的数据分析作业的处理时延。由于相比与云数据中心，边缘集群的处理能力相对较弱，因此式（2）中在边缘集群 $\phi(s)$ 的处理时延为 $\gamma_{\phi(s)} e_d^j$ ，其中 $\phi(s) = i$ ，表示计算单元 s 所处的边缘集群为 i； $\gamma_{\phi(s)}$ 为边缘集群 $\phi(s)$ 相对于云数据中心的处理速率比。对于数量为 $|T_i^j| - \sum_{d \in T_i^j, \phi(s)=i} I_{ds}^j$ 的数据分析任务而言，它们相互之间共享带宽 $B_{\phi(s)}^j$ ，因此总的传输时延是单个任务传输时延的 $|T_i^j| - \sum_{d \in T_i^j, \phi(s)=i} I_{ds}^j$ 倍。最后，在这些传输到数据中心的数据分析任务中，最后完成的是执行时延最长的任务。因此，在传输负载的基础上增加了这些数据分析任务中最长的执行时延。

1.2　跨域大数据处理作业调度问题

对于一个数据分析作业 j 来说，目标是最小化该作业的完成时间，而这取决于最晚完成的那个任务。由于相关任务分布于各个边缘集群或云数据中心，因此面向单个作业的任务调度目标可以转换为最小化任务所在各个集群中最大的负载，即

$$\max_s \{U_s^j, V_{\phi(s)}^j\} \tag{3}$$

就一系列作业而言，单独优化局部每个作业的完成时间并不一定能最小化作业的平均完成时间。为此，本文定义以优化系列作业平均完成时间的跨域大数据处理任务分配问题。

定义 1　跨域大数据处理作业任务分配问题 Geo-distributed big data analytics task assignment,Geo-TA）。针对某 DAG 型大数据处理作业的一批待执行任务，将这些任务指派到相关数据集群或转移到数据中心，以最小化所有作业的平均完成时间。每个作业的完成时间由式（3）定义。

$$Min: \frac{1}{n} \sum_j \max_s \{U_s^j, V_{\phi(s)}^j\} \tag{4}$$

$$s.t. \forall j, d: \sum_s I_{ds}^j = 1; I_{ds}^j = \{0,1\} \tag{5}$$

定理 1　跨域大数据处理作业任务分配问题 Geo-TA 是 NP 难问题。

证明　已知多处理器调度判定问题为 NP 难问题[13]，其定义为：给定 n 个处理器与 m 个作业，各作业的处理时间为 $p_1, \ldots, p_m$ ，要求判定是否存在一种调度 ψ ，使得在该调度下完成时间小于等于给定参数 k。即，将所有作业均指派到某一个处理器 $\forall i \in [1,m], \psi(i) \in [1,n]$ ，要求判定 $\max_s \{\sum_{i:\psi(i)=s} p_i\} \le k$ 是否成立。

对于任意一个多处理器调度判定问题的实例，都能将其在多项式时间内规约到一个 Geo-TA 判定问题的实例，并且该两个判定问题在任何调度策略下的输出一致。Geo-TA 判定问题的定义为，给定参数 k，Geo-TA 的整体完成时延，即式（4）是否小于等于 k。首先，在跨域资源部署 Geo-TA 的判定问题中，构建一个边缘集群拥有 n 个计算单元，且可用带

宽为一个无穷小值 B。并构造一个具有 m 个数据分析任务的数据分析作业。每个以数据块 i 作为输入数据的数据分析任务的执行时间对应于 $e_i^1 = p_i$。这样就能在 $O(n+m)$内从任意一个多处理器调度判定问题实例规约到跨域资源部署 Geo-TA 的判定问题实例。同时使得多处理器调度判定问题的参数 k 就是跨域资源部署 Geo-TA 判定问题的参数 k。该规约过程是多项式的。接着，对于任意一种多处理器调度策略，若其将作业 i 调度到处理器或是机器 s，那么在 Geo-TA 中就将第 i 号数据分析任务调度到计算单元 s（由于带宽过低，因此任务不会调度到云数据中心）。如此这样，多处理器调度判定问题中的总体完成时延为 $\max_s\{\sum_{i:\psi(i)=s} p_i\}$，与 Geo-TA 中的整体完成时间一致。所以在相同参数 k 下两个判定问题的输出一致。

那么，既然 Geo-TA 判定问题是 NP 完全问题，则 Geo-TA 问题是 NP 难问题。否则令 k 为任意比其最优解小的值，则 Geo-TA 判定问题能够在多项式时间内进行判定，与该判定问题是 NP 完全问题矛盾。

2　在线随机任务调度算法与捎带式数据重部署策略

由于上述任务调度问题为 NP 难问题，且作业依次不断到达，本文提出基于在线随机化的资源部署算法，试图最小化该作业的完成时间，并由此设计捎带式的数据重部署策略，并从理论上证明了在这样的任务调度与数据重部署机制下，系列作业的平均完成时间能以大概率聚集在最优解附近。

2.1　在线随机任务调度算法

最小化当前作业完成时间的任务调度问题，即最优化式（3），是 Geo-TA 问题的特例，事实上也是 NP 难问题，但该问题可以松弛转换为线性规划问题 lpGeo-TA (linear programming Geo-TA)。虽然线性规划求得的解不能直接应用于原跨域资源部署问题，但反映了算法对于当前资源部署的偏好。因此，本文利用该线性规划得到的理论最优解，并将其作为概率对数据分析任务进行调度。对于一系列数据分析作业来说也能够证明其结果以大概率稳定在最优解附近。

算法 1. 在线随机化任务调度算法 ranTA

1　求解 lpGeo-TA → { p_{ds}^j }
2　**for** 每一个任务 **do**
3　{ p_{ds}^j } 概率舍入成为 { I_{ds}^j }
4　以{ I_{ds}^j }调度该任务到计算单元 s
5 end for

算法 1 首先将原问题，即跨域资源调度问题松弛成为一个线性规划问题 lpGeo-TA（第 1 行）。在式（5）的基础上，以单个作业 j 为优化目标，将变量松弛成为[0, 1]的实数：

$$\forall d: \sum_s p_{ds}^j = 1; p_{ds}^j = [0,1] \tag{6}$$

由于该问题的变量为[0, 1]的实数，可用线性规划高效求解。对于每一个任务，对求得的{ p_{ds}^j }以概率舍入的方式（第 3 行）得到{ I_{ds}^j }。具体方式为，对于作业 j 的每一个数据分析任务，即其中一个以数据块 d 作为输入的任务，选取一个随机的小数 $r \in (0, 1]$。如果 r 落在 $(\sum_{k=0}^{s-1} p_{dk}^j, \sum_{k=0}^{s} p_{dk}^j]$，则 I_{ds}^j 为 1，否则为 0。这样的随机舍入策略能够保证对于任何一个数据分析任务，有且仅有一个计算单元能够服务该任务。同时，由于 I_{ds}^j 等于 1 的概率恰好为 r 落到区间 $(\sum_{k=0}^{s-1} p_{dk}^j, \sum_{k=0}^{s} p_{dk}^j]$ 的概率，I_{ds}^j 等于 1 的概率也即 p_{ds}^j。

最后，在进行真实数据分析任务调度的时候。可以先预先进行两次伪调度部署，取其中负载较低的一种方案作为真实部署数据分析任务的策略。这是因为，在进行了两次比较的选择后其中较低的这种策略能够使得高负载出现的概率进一步降低。算法第 1 行中由 lpGeo-TA 定义的问题能够使用线性规划技术进行高效求解；算法剩余部分（2~5 行）的复杂度仅为 O(ξ)，其中 ξ 为一个作业中包含的计算任务数目的上限。

2.2　捎带式数据重部署策略

在将任务调度至云数据中心后，可以立刻选择将数据留存下来（捎带式数据重部署）。这是因为任务的执行本身即需要获取数据。将任务调出至数据中心后可以直接利用已经传输的数据进行留存。这样后续的数据分析作业只要再一次需要该数据块，就能直接在数据中心进行执行，而不必从边缘计算集群再一次地传输数据。

与此同时，捎带式数据重部署还能够减轻边缘计算集群的负担，这是因为后续作业的数据分析任务将会在能力强大的云数据中心中进行计算，从而可以防止边缘计算集群成为热点与瓶颈，特别是在异构环境下，一些边缘计算集群的能力相对较弱，大量数据分析任务滞留本地将会对边缘计算集群产生极大的负载负担。

算法 2　捎带式数据重部署策略 *ran*TA-*data*

{ I_{ds}^j } ⟵ *ran*TA

$$\Omega = \max_i\{\max_{\phi(s)=i}\{U_s^j, V_i^j\}\}$$

for 每一个任务 **do**
if 该任务调度到云数据中心 **do**
将数据留存至数据中心
end if
end for
for 每一个边缘集群 i **do**
用最多 $\Omega - \max_i\{\max_{\phi(s)=i}\{U_s^j, V_i^j\}\}$ 进行数据重部署
end for

该算法与在线任务调度算法不同在于，其利用 *ran*TA 的输出结果进行数据重部署，即在进行任务调度后直接将关联的数据留存至云数据中心。正是因为该数据重部署策略无须额外代价，即第 3~7 行，因此称为捎带式数据重部署机制。此外，在进行捎带式数据重部署的过程中还可以利用在各个边缘集群中各异的任务完成时间进行优化，第 8-10 行。作业 j 关联的批量数据分析任务在各个边缘集群上的负载为 $\max_{\phi(s)=i}\{U_s^j, V_i^j\}$，正是由于在各个边缘集群上 $\max_{\phi(s)=i}\{U_s^j, V_i^j\}$ 的值可能出现差异，原因涉及边缘集群的计算能力、带宽、待处理负载及数据的分布。因此可以利用这样的负载差异，让数据能够在最滞后边缘集群完成前进行重部署，即在下式的限制内进行数据重部署 $\Omega - \max_i\{\max_{\phi(s)=i}\{U_s^j, V_i^j\}\}$。由于作业的完成取决于最滞后任务的完成，而最滞后任务必存在于某个最滞后边缘集群，因此只要在该边缘集群未完成任务前作业不会提前结束。利用该时间间隔进行数据重部署将会达到更好的效果。

2.3　理论分析

本节首先说明仅利用 ranTA 算法得到的数据分析作业完成时间能够以大概率稳定在其最优解的附近（定理 2）。且更

进一步，利用数据重部署策略 ranTA-data 能够达到更好的理论界（定理 3）。

定理 2 算法 1 得到的数据分析作业完成时间能够以大概率稳定在其最优解的附近，即

$$\frac{1}{n}\sum_j \max_s\{U_s^j, V_{\phi(s)}^j\} \le Opt + O(\frac{m}{n} + F(\frac{1}{\delta})) \quad (7)$$

证明 首先考察对于单个作业 j。以随机化的方式调度后，得到真实调度结果为 $\{U_s^j, V_{\phi(s)}^j\}$，下面构造随机变量

$$\Delta_s^d = \gamma_{\phi(s)} \sum_{x<d} (I_{xs}^j e_x^j - p_{xs}^j e_x^j) \quad (8)$$

用于刻画真实调度后的 U_s^j 与其期望负载之间的距离。这里需要说明的是 Δ_s^d 用于指示当数据块序号小于 d 的数据分析任务部署后，在计算单元 s 上产生的负载与其期望负载之间的距离。这里对数据块的枚举采用了其序号之间的大小关系。对于全局唯一的数据块序号，是可以进行大小比较的。可以得到

$$\forall s,d: E[\Delta_s^d] = E[\gamma_{\phi(s)} \sum_{x<d} (I_{xs}^j e_x^j - p_{xs}^j e_x^j)] = 0 \quad (9)$$

若作业 j 使用到的数据块中，数据块 d 的后继数据块序号为 $f(d)$，那么可以得到

$$|\Delta_s^{f(d)} - \Delta_s^d| = \gamma_{\phi(s)} |I_{f(d)s}^j e_{f(d)}^j - p_{f(d)s}^j e_{f(d)}^j|$$

$$\le \gamma_{\phi(s)} e_{f(d)}^j \quad (10)$$

也即序列{ Δ_s^d }为一个鞅差序列。将 Azuma[14]不等式应用到该鞅差序列上，就能得到：

$$\Pr[\Delta_s^d \ge t] \le \exp\{-\frac{t^2}{2\sum_{d\in T_{\phi(s)}^j} (\gamma_{\phi(s)} e_d^j)^2}\} \quad (11)$$

该不等式中 d 表示数据分析作业 j 所访问数据块中最大的序号。将上式展开后，可以得到：

$$\Pr[U_s^j - E[U_s^j] \ge t] \le \exp\{-\frac{t^2}{2\sum_{d\in T_{\phi(s)}^j} (\gamma_{\phi(s)} e_d^j)^2}\} \quad (12)$$

等价于以下不等式至少以(1-δ)概率成立，即将不等式右部分看成整体 δ：

$$U_s^j \le E[U_s^j] + \gamma_{\phi(s)} \max_{d\in T_{\phi(s)}^j}\{e_d^j\} \sqrt{2|T_{\phi(s)}^j| \ln\frac{1}{\delta}} \quad (13)$$

上式表示在使用了随机化调度后，在计算单元 s 上的负载距离其期望不会太远。且该距离随着概率(1-δ)的减少而指数增大。又由于 $E[U_s^j]$ 就是在利用了{ p_{ds}^j }得到的理论最优解，则上式意味着在进行随机化的调度后，在所有计算单元间都有 $\{U_s^j\}$ 以大概率稳定在其最优解附近，即以下不等式以至少(1-δ)概率成立：

$$\max_s\{U_s^j\} \le \max_s\{E[U_s^j]\} + \max_{i,d\in T_i^j}\{\gamma_i e_d^j\} \sqrt{2|T_i^j| \ln\frac{1}{\delta}} \quad (14)$$

另一方面，对于 V_i^j，同样可以进行类似的分析。可以得出以下不等式以至少(1-δ)概率成立：

$$V_i^j \le E[V_i^j] + \frac{\tau}{B_i^j} \sqrt{2|T_i^j| \ln\frac{1}{\delta}} \quad (15)$$

同样，由于 $E[V_i^j]$ 就是在利用了{ p_{ds}^j }得到的理论最优解，则上式意味着在进行随机化的调度后，在所有边缘集群中，{ V_i^j }以大概率稳定在其最优解附近，即以下不等式至少以(1-δ)概率成立：

$$\max_s\{V_{\phi(s)}^j\} \le \max_s\{E[V_{\phi(s)}^j]\} + \max_{\phi(s)}\{\frac{\tau}{B_{\phi(s)}^j}\} \sqrt{2|T_{\phi(s)}^j| \ln\frac{1}{\delta}} \quad (16)$$

那么在所有的边缘集群间就有以下不等式以(1-δ)概率成立：

$$\max_{\phi(s)=i}\{U_s^j, V_i^j\} \le z_j + \max_{i,d}\{\gamma_i e_d^j, \frac{\tau}{B_i^j}\} \sqrt{2|T_i^j|_{\max} \ln\frac{1}{\delta}} \quad (17)$$

其中不等式的左边部分即为真实的调度负载，右边的第一项为在各计算单元上调度期望的最大值，也即利用{ p_{ds}^j }进行调度的理论最优，这里用 z_j 代替。不等式的最后一部分代表着真实调度与最优局部调度之间的距离。且该距离随着概率(1-δ)的减少而指数级增大。不妨设所有作业之间式（17）最右边项的最大值为 $F(\frac{1}{\delta})$。那么对于所有的作业 j 来说，均有以下不等式以至少(1-δ)的概率成立：

$$\max_s\{U_s^j, V_{\phi(s)}^j\} \le z_j + F(\frac{1}{\delta}) \quad (18)$$

接下来考虑一系列作业。对于每个作业，式（18）不成立的概率至多为 δ。那么利用 Union Bound[15]，对于一系列 n 个作业，上式至少有一个不成立的概率至多为 nδ。也即对于系列作业，以下不等式以至少(1-nδ)成立。

$$\frac{1}{n}\sum_j \max_s\{U_s^j, V_{\phi(s)}^j\} \le \frac{1}{n}\sum_j (z_j + F(\frac{1}{\delta})) \quad (19)$$

又由于对于局部的理论最优解来说，其总是比任何整数调度的结果优，也因此 z_j 是作业 j 任何整数调度结果的下界。对于任何一个整数调度来说，其上界是将当前在边缘集群中的所有数据调度先传输到数据中心再进行执行。所以有以下不等式以至少(1-nδ)成立。

$$\frac{1}{n}\sum_j \max_s\{U_s^j, V_{\phi(s)}^j\} \le \frac{1}{n}(\sum_j (\max_d\{e_d^j\} + \max_i\{\frac{\tau|T_i^j|}{B_i^j}\} + nF(\frac{1}{\delta}))) \quad (20)$$

最后，由于全局的最优解不会好于在一开始所有的数据已经在云数据中心，且所有上传的数据量最多为总的数据访问量 m，因此，式（20）可以转换为以下不等式以至少(1-nδ)成立。

$$\frac{1}{n}\sum_j \max_s\{U_s^j, V_{\phi(s)}^j\} \le Opt + O(\frac{m}{n} + F(\frac{1}{\delta})) \quad (21)$$

也即，仅用算法 ranTA，跨域资源部署的结果以大概率稳定在其最优解附近。

定理 2 给出了应用 ranTA 算法，系列作业平均完成时间的理论上界。传统执行的过程中，即使数据被转移至云数据中心，任务完成后也不会留存。本文提出捎带式数据重部署中，转移至云数据中心的数据在任务执行完毕后将直接留存，且会根据作业特征，调度更多任务至云数据中心（即 ranTA-data），以优化后续作业在访问相同数据时收益。定理 3 给出了应用 ranTA-data 下，系列作业的平均完成时间上界。

定理 3 利用捎带式数据重部署策略 ranTA-data，得到的数据分析作业完成时间能够以大概率稳定在其最优解的附近，即（其中 m' 为所有数据访问中不同数据块的数目）：

$$\frac{1}{n}\sum_j \max_s\{U_s^j, V_{\phi(s)}^j\} \le Opt + O(\frac{m'}{n} + F(\frac{1}{\delta})) \quad (22)$$

证明 对于每一次调度，假设在进行作业 j 调度后，最优调度从边缘集群 i 向云数据中心传输的任务数量为 N_i^j，那么式（20）可以改写为

$$\frac{1}{n}\sum_j \max_s\{U_s^j, V_{\phi(s)}^j\} \le \frac{1}{n}(\sum_j (\max_d\{e_d^j\} + \max_i\{\frac{\tau(N_i^j+1)}{B_i^j}\} + nF(\frac{1}{\delta}))) \quad (23)$$

这是由于如果保持本地计算负载不变，再将多一个任务传输至云数据中心都会使得最优调度的整体时延变得更长（z_j是作业 j 任何整数调度结果的下界）。如果基于最优调度进行捎带式数据重部署，那么对于重复的数据而言，数据最多仅会上传一次，因此有

$$\sum_j \max_i \{N_i^j\} \le m' \tag{24}$$

所以式（23）变为

$$\frac{1}{n}\sum_j \max_s \{U_s^j, V_{\phi(s)}^j\} \le \frac{1}{n}(\sum_j (\max_d \{e_d^j\} + m' + n + nF(\frac{1}{\delta}))) \tag{25}$$

$$\le Opt + O(\frac{m'}{n} + F(\frac{1}{\delta}))$$

定理 2 与定理 3 的形式类似，区别在于 *m* 和 *m'*，其中 *m* 是任务相关的所有数据块数目，*m'*是任务相关的不相同数据块的数目。*m* 中包含着重复性的数据访问，因此，*m'*小于 *m*。直观上，得益于使用捎带式数据重部署，将数据留存至云数据中心为后续系列作业使用，ranTA-data 策略的理论上界优于 ranTA。

3　仿真实验

3.1　评价方法与设计

仿真实验对于算法的评价如同优化目标，即为系列作业的平均完成时间。本文将对这四个算法的性能进行比较：a）本地执行，直接将任务部署于数据所在的边缘集群；b）聚集[16]，将所有数据先汇聚到云数据中心再执行；c）*ranTA*，局部最优解进行调度，但任务执行后不将从边缘转移来的数据保存在云数据中心；d）*ranTA-data*，在线调度且云数据中心保存由边缘传来的数据。

仿真实验模拟了云-边缘场景下常见的参数设置：

a）云—边缘环境设定。800 个边缘集群，一个中心云数据中心；每个边缘集群拥有 10-150 个计算单元；边缘集群相比云数据中心的处理速度比为 1~5 倍，且每个边缘集群的带宽变化范围为 100Mbps-1Gbps[5]。

b）数据分析作业。100 个数据分析作业，每个数据分析作业的数据分析任务为 50-750 个[17]，每个数据分析任务所处理的数据块大小为 64MB。

c）数据分布。30000 个数据块以 Zipf 分布[9]，默认参数为 0.85，部署在不同的边缘集群内；同时为了体现数据的重复使用特性[18]，有 30%的数据会以 0.8 的概率进行访问。

3.2　实验结果分析

图 1~3 展示了当云-边缘环境设定发生变化时作业完成时间的变化。其中，10%↓表示计算单元数目或带宽较适度的参数设置有 10%的下降。ranTA-data 表现最好，相比与传统的数据本地性和聚集策略能够在计算单元数目、带宽和边缘集群数目发生变化时分别平均提升 33.9%、31.6%和 24.3%。相比不进行数据重部署的 ranTA 在计算单元数目、带宽和边缘集群数目发生变化时分别平均提升 14%，14.3%和 16%。

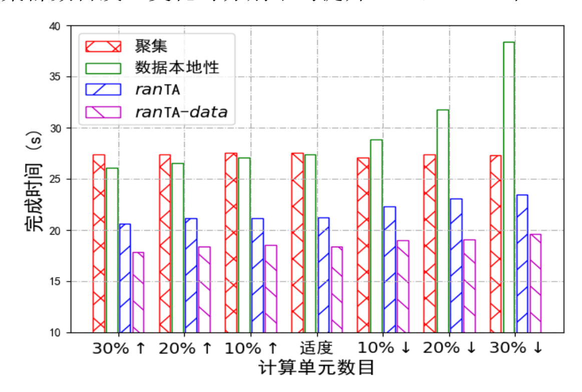


图 1　完成时间随计算单元数目的变化

Fig. 1　Results under various settings on compute slot.

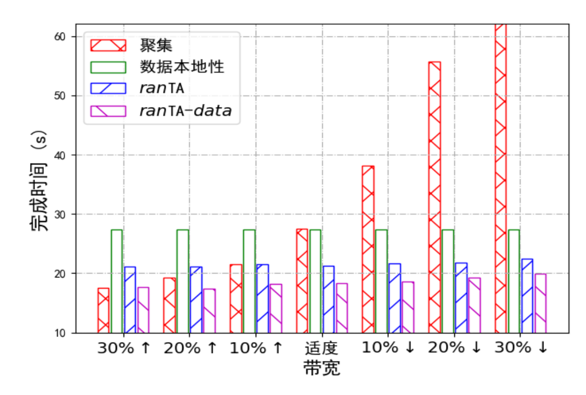


图 2　完成时间随带宽的变化

Fig. 2　Results under various settings on bandwidth.

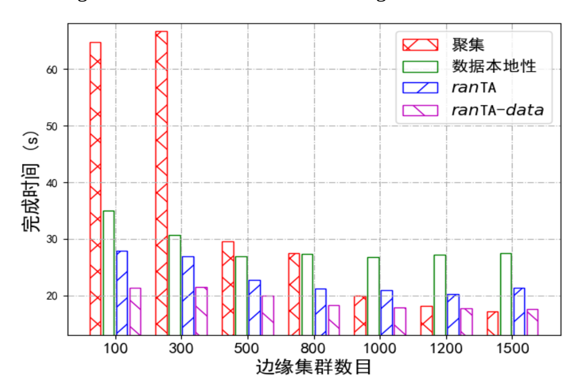


图 3　完成时间随边缘集群数目的变化

Fig. 3　Results under various settings on the number of edges.

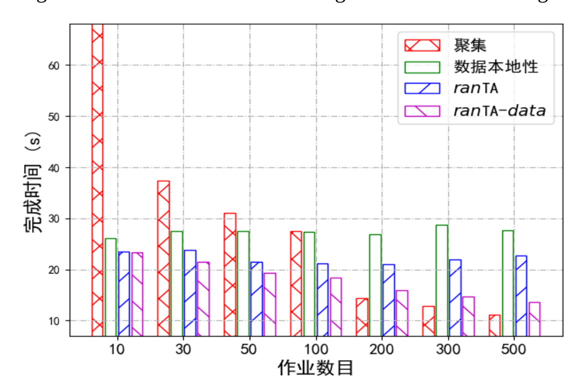


图 4　完成时间随作业数目的变化

Fig. 4　Results under various settings on the number of jobs.

图 1 和 2 表明当边缘集群资源相对较少时（计算单元数目或是带宽）ranTA 能够相比其他两个算法提升更大。这是因为 ranTA 在每一个作业到达的时候均会结合当下的系统状态将部分任务调度到云数据中心中去。使得本地或是网络带宽的高负载得以缓解，避免在本地进行长时间的排队等待空闲计算资源或是和大量传输任务一起共享稀缺网络带宽。但是相比于 ranTA-data，其提升仍然有限，这是因为 ranTA-data 通过捎带式的数据重部署，会将所有调度到云数据中心的数据进行留存。这样，只要后续的作业再一次使用到该数据，就能直接收益。图 3 展示了当边缘集群数目较少的时候，由于数据分布的聚集，导致大量数据聚集在少数边缘集群内，给相应的边缘集群增加了负担。因此，聚集和数据本地性策略都居高不下。虽然 ranTA 在该场景下的完成时间相对也高，但是也已经尽力进行负载的疏散。ranTA-data 表现最好，除了转移负载外，保存在云数据中心的数据会有效提升后续作业性能。

图 4 和 5 展示了当数据分析作业设定发生变化时作业完

成时间的变化。ranTA-data 依然表现出色，相比与传统的数据本地性和聚集策略能够在作业数目和平均任务数目发生变化时分别平均提升 31.6%和 34.1%。相比不进行数据重部署的 ranTA 在作业数目和平均任务数目发生变化时分别平均提升 18%和 16%。当作业数目不断增大或是作业包含的数据分析任务不断增加的时候，给边缘集群带来的负载不断增加，因此聚集策略和数据本地性策略的完成时间均会大幅度增加。但是 ranTA-data 增加缓慢，这是因为 ranTA-data 在运行时刻就不断进行数据分析任务的负载均衡，并且数据的转移也为后续作业服务。

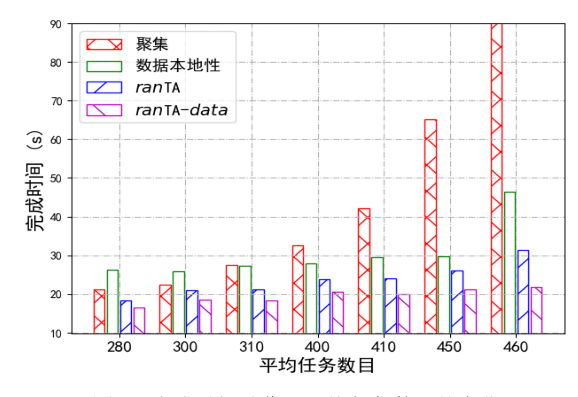


图 5　完成时间随作业平均任务数目的变化

Fig. 5　Results under various settings on average number of tasks within a job.

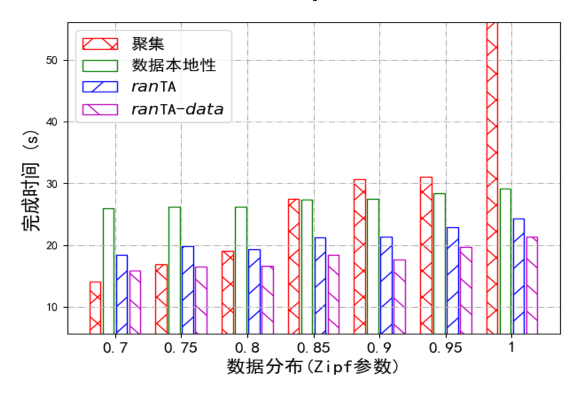


图 6　完成时间随数据分布的变化

Fig. 6　Results under various settings on data distribution.

图 6 展示了随数据分布变化而变化的平均作业完成时间。ranTA-data 相比与传统的数据本地性和聚集策略提升平均提升 27.2%，相比仅在线数据分布的 ranTA 平均提升至少 13%。数据分布中 Zipf 参数越大，意味着数据越不均匀，越集中于少数边缘集群内。因此对部分边缘集群的高负载使得聚集策略和数据本地性策略产生极高时延。当数据分布越均匀时，由于边缘集群的数目巨大，导致在极个别边缘集群上出现 ranTA-data 略差于将所有数据上传的情况。这是由于 ranTA 和 ranTA-data 均是基于随机化的调度策略，当数据分析数目较少时，采用随机舍入策略可能使得完成时间差于将所有任务调度到云数据中心。最后，数据本地性在数据分布极为均匀的情况下差于聚集策略是因为边缘集群相比云数据中心有数据处理的速度比。使得即使同样的任务，全部在本地执行仍然要比全部上传云端代价高。

4　结束语

本文针对跨域环境下，在异构边缘集群间进行跨域大数据处理时容易使得计算能力较弱或是稀缺带宽连接的边缘集群成为瓶颈，提出了在线随机化任务调度算法 ranTA 和捎带式数据重部署策略 ranTA-data，以系统偏好作为概率调度每个计算任务，并在此基础上提出了捎带式数据重部署策略，并证明在应用该调度机制后，作业的平均完成时间能以大概率聚集在最优解附近。该方法使得系列作业的平均完成时间得以降低，具有重要的理论和实际应用意义。

参考文献：

[1] Calder M, Fan Xun, Hu Zi, *et al.* Mapping the expansion of Google's serving infrastructure [C]// Proc of Conference on Internet Measurement. New York: ACM Press, 2013: 313-326.

[2] Jalaparti V, Bodik P, Menache I, *et al.* Network-aware scheduling for data-parallel jobs: plan when you can[C]// Proc of ACM Conference on Special Interest Group on Data Communication. New York: ACM Press, 2015: 407-420.

[3] 卢慧,高弘博,张丰满,等.Hadoop 云平台下基于资源预估的作业调度算法 [J]. 计算机应用研究,2016,33(8): 2311-2314. (Lu Hui, Gao Hongbo, Zhang Fengman,*et al.* Job scheduling algorithm based on data-aware in Hadoop [J]. Application Research of Computers, 2016, 33(8):2311-2314.)

[4] Vulimiri A, Curino C, Godfrey P B, *et al.* Global analytics in the face of bandwidth and regulatory constraints[C]//Proc of the 12th USENIX Symposium on Networked System Design and Implementation. Berkeley, CA: Usenix Accociation, 2015: 323-336.

[5] Pu Qifan, Ananthanrayanan G, Bodik P, *et al.* Low latency Geo-distributed data analytics [C]// Proc of ACM Conference on Special Interest Group on Data Communication. New York: ACM Press, 2015: 421-434.

[6] Viswanathan R,Ananthanarayanan G , Akella A. CLARINET: WAN-aware optimization for analytics queries[C]// Proc of the 12th USENIX Symposium on Operating Systems Design and Implementation. Berkeley, CA: Usenix Accociation, 2016: 435-450.

[7] Yu Boyang,Pan Jianping. Location-aware associated data placement for geo-distributed data-intensive applications[C]//Proc of IEEE INFOCOM. Piscataway, NJ: IEEE Press, 2015: 603-611.

[8] Hu Zhiming, Li Baochun,Luo Jun. Flutter: scheduling tasks closer to data across geo-distributed datacenters [C]//Proc of IEEE INFOCOM. Piscataway, NJ: IEEE Press, 2016: 1-9.

[9] Hung C C, Golubchik L,Yu Minlan. Scheduling jobs across geo-distributed datacenters [C]//Proc of the 6th ACM Symposium on Cloud Computing. New York: ACM Press, 2015: 111-124.

[10] Jin Yibo, Qian Zhuzhong, Guo Song, *et al.* ran-GJS: orchestrating data analytics for heterogeneous geo-distributed edges [C]// Proc of the 47th International Conference on Parallel Processing. New York: ACM Press, 2018: 29: 1-29: 10.

[11] Ghemawat S, Gobioff H and Leung S. The Google file system [C]// Proc of the 19th ACM Symposium on Operating Systems Principles. New York: ACM Press, 2003: 29-43.

[12] 曹书豪, 张昌宏, 麻旻.一种改进的 Hadoop 多用户作业调度方法 [J]. 计算机应用研究 ,2015,32(5):1395-1398. (Cao Shuhao, Zhang Changhong,Ma Min. Improved method in solving Hadoop multi-user scheduling [J]. Application Research of Computers,2015,32(5): 1395-1398.)

[13] Chen Jianer,Lee C Y. General multiprocessor task scheduling [J]. Naval Research Logistics, 1999, 46 (1): 57-74.

[14] Azuma K. Weighted sums of certain dependent random variables [J]. Tohoku Mathematical Journal, 1967, 19 (3): 357-367.

[15] Galamnos J,Simonelli I. Boneferroni-Type inequalities with applications, probability and its applications [M]. New York: Spring-Verlag, 1996.

[16] Rabkin A, Arye M, Sen S, *et al.* Aggregation and degradation in jetstream: streaming analytics in the wide area[C]//Proc of the 11th USENIX Symposium on Networked System Design and Implementation. Berkeley, CA: USENIX Accociation, 2014: 275-288.

[17] Ananthanarayanan G, Hung Chienchun, Ren Xiaoqi, et al. GRASS: trimming stragglers in approximation analytics [C]//Proc of the 11th USENIX Symposium on Networked System Design and Implementation. Berkeley, CA: USENIX Accociation, 2014: 289-302.

[18] Chen Yanpei, Archana G, Rean G, et al. The Case for evaluation MapReduce performance using workloads suites [C]// Proc of the 19th Annual IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication System. Washington DC: IEEE Computer Society, 2011: 390-399.